

Toward a Categorical Foundation for Energetically Coherent Computation

Energy-Aware Functors, Adjoint Compilation, and the
Thermodynamic Redefinition of Correctness

Daniel Scott Matthews d@3-e.net

April 13, 2026

Abstract

We propose a categorical framework in which energy flow is a first-class constituent of computation, not an extrinsic engineering constraint. Standard software architecture implicitly treats composition as free, duplication as free, and energetic cost as a scalar annotation appended after the fact. We argue that this assumption is not merely suboptimal but *categorically ill-formed*: the realization functor from an algorithmic category $\mathcal{A}$ to a physical category $\mathcal{P}$ is, in current systems, neither cost-preserving nor structure-preserving with respect to energy.

We introduce the notion of an *energetically coherent adjunction* $F \dashv G$ between $\mathcal{A}$ and $\mathcal{P}$, interpret the unit and counit as expressing optimal tension between algorithmic structure and physical realization, and show that the separation-of-concerns principle is precisely a *forgetful functor* that discards energetic structure without providing a right adjoint by which it can be recovered.

We further argue that the abstraction functor $G : \mathcal{P} \rightarrow \mathcal{A}$ is generically non-unique—making a profunctor formulation more faithful than a functor—and identify three underappreciated consequences: (i) the interaction term in compositional energy profiles undergoes a phase transition at large compositional depth that defeats purely local correction schemes; (ii) Landauer’s principle entangles the energetic structure of a computation with its information-theoretic structure at the level of individual morphisms; and (iii) the bootstrapping problem in fine-grained energy measurement is not merely experimental noise but a structural feature of the measurement relation itself.

We conclude that the appropriate notion of *correctness* for a computation is not satisfaction of an input-output specification alone, but co-satisfaction of that specification *and* energetic coherence within a controlled tolerance of the adjunction’s unit-counit pair.

Keywords: category theory, energy-aware computing, adjoint functors, linear logic, Landauer’s principle, profunctors, thermodynamics of computation, neuromorphic computing, compiler design, resource-sensitive semantics.

1 Introduction

The dominant paradigm in software engineering is one of *layered abstraction*: each layer of a system presents a clean interface to the layer above while hiding implementation details below. This principle—separation of concerns—has been enormously productive. It has made large-scale collaborative software development possible, enabled formal verification at bounded scopes, and decoupled the pace of algorithmic progress from the pace of hardware change.

Yet the paradigm embeds a hidden assumption: that *composition is free*. In the mathematical language of category theory, this corresponds to working in a Cartesian category—one in which morphisms can be freely copied, discarded, and composed without incurring structural cost. This is a faithful model of mathematical functions. It is a *poor* model of physical processes.

Every state transition in a physical computer is a thermodynamic event. Every memory fetch dissipates energy; every logical gate switched generates heat; every bit erased—by Landauer’s principle [Landauer, 1961]—contributes irreversibly to entropy. The gap between the Cartesian abstraction and physical reality is not merely an engineering inconvenience. It is a *categorical mismatch* between the structure of the model and the structure of the substrate.

This paper is an attempt to make that mismatch precise, and to sketch the architecture of a framework in which it is resolved—not by discarding abstraction, but by demanding that abstractions preserve energetic structure up to controlled error.

1.1 The core thesis

We argue for three related claims, in increasing order of strength:

- (i) **Mismatch.** The realization functor $F : \mathcal{A} \rightarrow \mathcal{P}$ from an algorithmic category to a physical category is, in current systems, neither cost-preserving nor compositionally faithful with respect to energy. This is not a deficiency of particular implementations but a structural consequence of the Cartesian assumption built into standard programming abstractions.
- (ii) **Adjunction.** The appropriate relationship between $\mathcal{A}$ and $\mathcal{P}$ is not a one-way compilation map but an *adjunction* $F \dashv G$, where $G : \mathcal{P} \rightarrow \mathcal{A}$ is a lifting functor from physical measurement to algorithmic structure. The unit and counit of this adjunction encode the optimal tension between algorithmic design and physical realization.

- (iii) **Correctness.** A computation is correct not merely if its outputs match a specification, but if its *energetic trajectory* remains within a controlled neighborhood of the trajectory predicted by the adjunction. This is a strictly stronger notion of correctness, reducible to the classical notion only in the Cartesian limit where energy is free.

1.2 Structure of the paper

Section 2 establishes the categorical background. Section 3 defines the algorithmic and physical categories formally. Section 4 develops the adjunction and its interpretation. Section 5 recasts separation of concerns as a forgetful functor. Section 6 argues for the profunctor generalization of G . Section 7 analyzes the interaction term and its phase-transition behavior. Section 8 connects the framework to Landauer’s principle and the thermodynamics of computation. Section 9 addresses the bootstrapping problem in energy measurement. Section 10 reformulates correctness. Section 11 grounds the framework in concrete cases. Section 12 surveys related work, and Section 14 concludes with directions for further research.

2 Categorical Background

We assume familiarity with basic category theory at the level of [Mac Lane, 1998, Riehl, 2017]. We recall here only the structures central to our development.

Definition 2.1 (Symmetric Monoidal Category). A *symmetric monoidal category* (SMC) is a category $\mathcal{C}$ equipped with a tensor product $\otimes : \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{C}$, a unit object I , and natural isomorphisms expressing associativity, left/right unit, and symmetry, satisfying the standard coherence conditions [Mac Lane, 1998].

SMCs generalize Cartesian categories by dropping the assumption that the diagonal map $\Delta_A : A \rightarrow A \otimes A$ and the terminal map $\epsilon_A : A \rightarrow I$ exist. In a Cartesian category, morphisms may be freely copied (Δ) and discarded (ϵ); in a general SMC, they may not.

Definition 2.2 (Resource-Sensitive Category). A *resource-sensitive category* is an SMC in which morphisms represent *processes that consume their inputs*: copying and discarding are not assumed. Morphisms compose sequentially via $\circ$ and in parallel via $\otimes$, and neither operation is free of cost.

This aligns with the categorical semantics of *linear logic* [Girard, 1987], in which propositions are resources consumed by proofs.

Definition 2.3 (Adjunction). An *adjunction* $F \dashv G$ between categories $\mathcal{C}$ and $\mathcal{D}$ consists of functors $F : \mathcal{C} \rightarrow \mathcal{D}$ and $G : \mathcal{D} \rightarrow \mathcal{C}$ together with a natural isomorphism:

$$\mathrm{Hom}_{\mathcal{D}}(F(c), d) \cong \mathrm{Hom}_{\mathcal{C}}(c, G(d))$$

natural in $c \in \mathcal{C}$ and $d \in \mathcal{D}$. The unit $\eta : \text{id}_{\mathcal{C}} \Rightarrow G \circ F$ and counit $\varepsilon : F \circ G \Rightarrow \text{id}_{\mathcal{D}}$ satisfy the triangle identities.

Definition 2.4 (Profunctor). A *profunctor* $H : \mathcal{C} \nrightarrow \mathcal{D}$ is a functor $H : \mathcal{C}^{\text{op}} \times \mathcal{D} \rightarrow \mathbf{Set}$. Profunctors generalize functors and represent heterogeneous relations between categories. Every functor $F : \mathcal{C} \rightarrow \mathcal{D}$ induces profunctors $\mathcal{D}(F-, =)$ and $\mathcal{D}(-, F=)$.

3 The Algorithmic and Physical Categories

We now define the two principal objects of study.

Definition 3.1 (Algorithmic Category $\mathcal{A}$). The *algorithmic category* $\mathcal{A}$ is a symmetric monoidal category whose:

- **Objects** are *data types* (or more generally, abstract resource types).
- **Morphisms** $f : A \rightarrow B$ are *computational processes* specified intensionally (programs, algorithms, or equivalence classes thereof under some chosen notion of behavioral equivalence).
- **Composition** $g \circ f$ represents sequential execution.
- **Tensor product** $A \otimes B$ represents parallel or independent resource combination.

$\mathcal{A}$ is *not* assumed Cartesian. The distinction between linear and non-linear (classical) computation is captured by whether the diagonal and weakening maps exist for a given object.

Definition 3.2 (Physical Category $\mathcal{P}$). The *physical category* $\mathcal{P}$ is a symmetric monoidal category whose:

- **Objects** are *physical states* of a computational substrate (configurations of charge, magnetic domain, spin, etc.).
- **Morphisms** $\phi : S \rightarrow S'$ are *physically realized transitions*: time-parameterized trajectories through state space, constrained by the laws of thermodynamics.
- **Composition** $\psi \circ \phi$ represents sequential physical process.
- **Tensor product** $S \otimes S'$ represents spatially or logically parallel physical process.

Definition 3.3 (Energy Profile Functor). An *energy profile* is a functor

$$E : \mathcal{P} \rightarrow \mathcal{P}\nabla\wr\{$$

where $\mathcal{P}\nabla\wr\{$ is a category of time-indexed energy distributions. For a morphism $\phi \in \text{Hom}_{\mathcal{P}}(S, S')$, $E(\phi)$ encodes:

- (a) the total energy dissipated;
- (b) the time-dependence of that dissipation;
- (c) locality properties (spatially local vs. non-local dissipation);
- (d) concurrency interactions when ϕ is composed in parallel.

Critically, the composition law for energy profiles is:

$$\mathbf{E}(\psi \circ \phi) = \mathbf{E}(\psi) \otimes \mathbf{E}(\phi) \otimes \Gamma(\phi, \psi)$$

where $\Gamma(\phi, \psi)$ is the *interaction term*: the energetic overhead arising from the physical coupling between the sequential processes ϕ and ψ . This term is not recoverable from $\mathbf{E}(\phi)$ and $\mathbf{E}(\psi)$ individually.

Remark 3.4. The interaction term Γ is the point at which the naïve additive model of energy cost—common in power-aware computing literature—breaks down. We return to this in Section 7.

4 The Energetically Coherent Adjunction

We are now in a position to state the central structural claim.

Definition 4.1 (Realization and Abstraction Functors). A *realization functor* is a functor $F : \mathcal{A} \rightarrow \mathcal{P}$ that maps each algorithm to its physical implementation and each algorithmic composition to the corresponding physical process sequence.

An *abstraction functor* is a functor $G : \mathcal{P} \rightarrow \mathcal{A}$ that lifts measured physical behavior to an inferred algorithmic structure.

Definition 4.2 (Energetically Coherent Adjunction). An adjunction $F \dashv G$:

$$\begin{array}{ccc} \mathcal{A} & \xrightarrow{F} & \mathcal{P} \\ & \xleftarrow{G} & \end{array}$$

is *energetically coherent* if the natural isomorphism

$$\mathrm{Hom}_{\mathcal{P}}(F(a), p) \cong \mathrm{Hom}_{\mathcal{A}}(a, G(p))$$

is compatible with the energy profile functor $\mathbf{E}$ in the following sense: the bijection maps *low-energy realizations* of a to *structurally tight abstractions* of p , and vice versa.

Proposition 4.3 (Interpretation of Unit and Coint). *Under an energetically coherent adjunction:*

- The unit $\eta_a : a \rightarrow G(F(a))$ expresses the information lost when an algorithm is realized and then re-abstracted from its physical execution: it is a measure of how much structural information the physical substrate fails to preserve.
- The counit $\varepsilon_p : F(G(p)) \rightarrow p$ expresses the energetic excess when a physically observed behavior is given an algorithmic interpretation and then re-realized: it measures how much energy the re-implementation wastes relative to the original.

The triangle identities then encode the consistency between these two losses: they cannot both be zero unless F and G are equivalences.

Remark 4.4 (Optimal Tension). What is colloquially described as the “optimal tension” between algorithmic design and physical realization is precisely the condition that the unit and counit be *small* in an appropriate metric—not zero, which would require the impossible, but minimized subject to the constraints imposed by the computational task and the physical substrate. The adjunction provides the formal language in which this minimization can be stated.

5 Separation of Concerns as a Forgetful Functor

We now give the categorical analysis of why standard layered architectures incur an irrecoverable energetic deficit.

Definition 5.1 (Forgetful Realization). A realization functor $F : \mathcal{A} \rightarrow \mathcal{P}$ is *energetically forgetful* if it does not preserve the energy profile structure: that is, if the diagram

$$\begin{array}{ccc} \mathcal{A} & \xrightarrow{F} & \mathcal{P} \\ & \searrow & \downarrow \varepsilon \\ & & \mathcal{P} \nabla \{ \end{array}$$

does not commute up to natural isomorphism. Equivalently, F is forgetful if the energy profile of $F(f \circ g)$ is not recoverable from the profiles of $F(f)$ and $F(g)$ and the abstract structure of f and g .

Theorem 5.2 (Separation of Concerns is Forgetful). *Any architecture that strictly enforces the separation-of-concerns principle via abstract interfaces induces an energetically forgetful realization functor.*

Proof sketch. Strict interface abstraction requires that the behavior observable at any layer boundary is fully determined by the interface specification, independent of the internal implementation. But energy profiles of composed operations depend on the internal interaction term $\Gamma(\phi, \psi)$, which is a function of the *implementation* of both ϕ and ψ , not merely their interface types. Hence the interface abstraction discards precisely the information needed to compute energy profiles compositionally. $\square$

Corollary 5.3. *An energetically forgetful F has no right adjoint G by which the discarded structure can be recovered: there is no functor $G : \mathcal{P} \rightarrow \mathcal{A}$ such that the natural isomorphism of Definition 4.2 holds. The information loss is permanent.*

Remark 5.4. This does not mean separation of concerns is harmful. It means that it is a *lossy compression* of the full computational structure—useful precisely because it reduces dimensionality, but irreversible. The deficit manifests as the persistent inability of high-level profiling to explain or predict fine-grained energy behavior without access to implementation internals.

6 The Abstraction Functor is Generically a Profunctor

A central assumption of Section 4 is that $G : \mathcal{P} \rightarrow \mathcal{A}$ is well-defined as a functor. We now argue that this assumption fails generically, and that the appropriate replacement is a *profunctor*.

Proposition 6.1 (Non-Uniqueness of Abstraction). *For a generic physical execution trajectory $p \in \mathcal{P}$, there exist multiple distinct algorithmic structures $a_1, a_2 \in \mathcal{A}$ such that $F(a_i) \cong p$ up to energetically negligible perturbation.*

The proposition follows from the straightforward observation that many distinct algorithms can produce identical energy signatures on identical hardware under identical workloads—a fact well-known to practitioners of power side-channel analysis [Kocher et al., 1999].

Definition 6.2 (Abstraction Profunctor). In place of a functor $G : \mathcal{P} \rightarrow \mathcal{A}$, we define an *abstraction profunctor*

$$\mathbb{G} : \mathcal{P} \nrightarrow \mathcal{A}$$

where $\mathbb{G}(p, a)$ is the set (or space) of *evidence structures* witnessing that physical trajectory p is a valid realization of algorithm a , subject to a specified energetic tolerance $\epsilon > 0$.

Remark 6.3 (Design-Decision Interpretation). The non-uniqueness of G is not merely a technical inconvenience. The choice of which algorithmic preimage to assign to a measured physical trajectory is a *design decision* that encodes assumptions about what computational structure is relevant to the designer’s purposes. Making this choice explicit via the profunctor—rather than hiding it in an underdetermined functor—is a form of intellectual honesty about the limits of reverse-engineering.

The adjunction condition must therefore be restated in profunctor terms using the calculus of coends [Lorègian, 2021]:

$$\int^{p \in \mathcal{P}} \text{Hom}_{\mathcal{P}}(F(a), p) \times \mathbb{G}(p, a') \cong \text{Hom}_{\mathcal{A}}(a, a')$$

This replaces the sharp bijection with a *weighted* correspondence between realizations and abstractions.

7 The Interaction Term and Its Phase Transition

We return to the interaction term $\Gamma(\phi, \psi)$ introduced in Definition 3.3 and analyze its behavior as a function of compositional depth.

Definition 7.1 (Compositional Depth). The *compositional depth* $d(f)$ of a morphism $f \in \text{Hom}_{\mathcal{A}}(A, B)$ is the minimum number of atomic morphisms whose composition equals f in any normal form.

Proposition 7.2 (Local Regime). *When $d(f)$ and $d(g)$ are both small, the interaction term satisfies*

$$\|\Gamma(F(f), F(g))\| \ll \|\mathbf{E}(F(f))\| + \|\mathbf{E}(F(g))\|$$

and the energy profile is well-approximated by the additive model $\mathbf{E}(F(f)) \otimes \mathbf{E}(F(g))$. Standard power-aware computing models operate in this regime.

Proposition 7.3 (Phase Transition at Large Depth). *At large compositional depth, cache-capacity saturation, NUMA (Non-Uniform Memory Access) effects, thermal throttling, and branch-predictor state divergence combine to produce interaction terms that satisfy:*

$$\|\Gamma(F(f), F(g))\| \sim \|\mathbf{E}(F(f))\| + \|\mathbf{E}(F(g))\|$$

In this regime, the interaction term is no longer a perturbative correction but a dominant contribution. The energy profile of the composed morphism is not recoverable from the profiles of its components by any local correction scheme.

Remark 7.4. This phase transition is the categorical explanation for a commonly observed empirical phenomenon: micro-benchmarked energy profiles of individual functions often fail to predict the energy profile of a complete application, even when the application is architecturally simple. The failure is not measurement error; it is the transition from the local to the non-local regime.

Definition 7.5 (Interaction Homology). We introduce, tentatively, the concept of *interaction homology*: a functor

$$H_* : \mathcal{P} \rightarrow \mathbf{GradedAb}$$

that assigns to each physical process a graded abelian group whose k -th degree captures k -fold interaction effects among composed sub-processes. The degree-0 piece recovers the additive model; higher degrees capture the interaction terms systematically. A full development of this construction is left for future work.

8 Landauer’s Principle and Morphism-Level Thermodynamics

The framework developed so far treats energy as a classical resource flowing through a computational graph. We now show that a deeper constraint is operative at the level of individual morphisms.

Theorem 8.1 (Landauer’s Principle, Landauer, 1961). *Any logically irreversible computation—specifically, any operation that erases information—must dissipate at least $k_B T \ln 2$ of energy per bit erased into the environment, where k_B is Boltzmann’s constant and T is the ambient temperature.*

Proposition 8.2 (Morphism-Level Consequence). *In the physical category $\mathcal{P}$, a morphism $\phi : S \rightarrow S'$ is thermodynamically reversible if and only if the corresponding algorithmic morphism $G(\phi) : G(S) \rightarrow G(S')$ (or its profunctor generalization) is injective in the algorithmic category. Non-injective algorithmic morphisms—projections, conditionals where one branch is taken and its alternative erased, compression operations—correspond to morphisms in $\mathcal{P}$ that obligatorily dissipate energy.*

Corollary 8.3 (Information-Energy Entanglement). *The energetic structure of an algorithm is entangled with its information-theoretic structure at the level of individual morphisms. A cost-preserving functor F must therefore track not merely energy flow but entropy production per morphism—and the entropy production is determined by the injectivity structure of the algorithm, not by its execution time or transistor count.*

Remark 8.4 (Connection to Reversible Computing). Bennett’s reversible computing framework [Bennett, 1973] and the subsequent thermodynamics-of-computation programme [Wolpert, 2019] demonstrate that Landauer’s bound is achievable in principle: any classical computation can be performed reversibly by augmenting the output with a “garbage” register that records erased information. In categorical terms, this corresponds to *extending* F so that every morphism in the image is injective—at the cost of expanding the object space of $\mathcal{P}$. The trade-off between energy minimization and resource (space) usage is precisely the adjunction-theoretic trade-off between F and G .

9 The Bootstrapping Problem in Energy Measurement

The profunctor $\mathbb{G}$ and the interaction term Γ both require fine-grained energy measurements as inputs. This creates a structural difficulty.

Proposition 9.1 (Measurement Context-Contamination). *Fine-grained measurement of the energy profile $E(F(f))$ for an isolated morphism f alters the execution context—specifically, the cache state, branch predictor history, and thermal envelope—in which any subsequent morphism g will execute. Consequently:*

$$E(F(f \circ g)) \neq E^{\text{meas}}(F(f)) \otimes E^{\text{meas}}(F(g)) \otimes \Gamma$$

where the superscript “meas” denotes the profile measured in isolated benchmarking. The discrepancy is not noise; it is a structural feature of the measurement relation.

Proof sketch. Isolated measurement of $F(f)$ warms instruction caches, pre-populates translation lookaside buffers, and shifts the branch predictor state. When $F(g)$ subsequently executes in the composed run, these are already warm—the execution context is not the same as the isolated measurement context. The difference is a systematic bias, not a random error. $\square$

Definition 9.2 (Contextual Energy Distribution). We replace the pointwise energy profile with a *contextual energy distribution*: a probability measure

$$\mu_f : \mathcal{C} \backslash \sqcup \} \sqcup \rightarrow \Delta(\mathcal{P} \nabla \{ \}$$

where $\mathcal{C} \backslash \sqcup \} \sqcup$ is a category of execution contexts and $\Delta(\mathcal{P} \nabla \{ \})$ is the space of probability distributions over energy profiles. The relevant observable for functoriality testing is the *marginal* distribution integrated over a representative context distribution, not the point estimate at any particular context.

Remark 9.3 (Experimental Design Consequence). Any empirical test of whether the energy composition law is functorial must therefore be *statistical*: it must sample over a distribution of execution contexts and test whether the compositional energy distribution matches the convolution of the component distributions (plus interaction term) in a distributional sense. Single-run benchmarks are insufficient by construction.

10 A Thermodynamic Redefinition of Correctness

We are now in a position to state the paper’s most consequential claim.

Definition 10.1 (Classical Correctness). A program $f \in \text{Hom}_{\mathcal{A}}(A, B)$ is *classically correct* with respect to a specification $\sigma : A \rightarrow B$ if there exists a natural transformation $f \Rightarrow \sigma$ in the appropriate functor category, or equivalently if the observable input-output behavior of f satisfies the denotational semantics prescribed by σ .

Definition 10.2 (Energetically Coherent Correctness). A program $f \in \text{Hom}_{\mathcal{A}}(A, B)$ is *energetically coherently correct* with respect to specification σ and energy tolerance ϵ if:

- (i) f is classically correct with respect to σ ; and
- (ii) the energy profile $E(F(f))$ satisfies

$$d(E(F(f)), E^*(F(\sigma))) \leq \epsilon$$

where d is a metric on $\mathcal{P}\nabla\{\}$, and $E^*(F(\sigma))$ is the *adjunction-optimal energy profile* determined by the unit-counit pair of $F \dashv G$.

Theorem 10.3 (Reduction to Classical Correctness). *In the Cartesian limit—where composition is free, duplication is free, and energy is abundant—the adjunction-optimal profile assigns zero cost to all morphisms, the tolerance condition $d \leq \epsilon$ is vacuously satisfied for all $\epsilon > 0$, and energetically coherent correctness reduces to classical correctness.*

Remark 10.4. Definition 10.2 implies that the same algorithm can be correct in one deployment context (abundant energy, high tolerance ϵ) and incorrect in another (resource-constrained embedded system, low tolerance). This is not a defect of the definition—it faithfully reflects engineering reality. The novelty is that this context-dependence is now a *formal* part of the correctness condition rather than an informal engineering judgment appended after verification.

11 Concrete Examples

11.1 The Fast Fourier Transform as Energetic Coherence

The Cooley-Tukey FFT algorithm [Cooley and Tukey, 1965] reduces the arithmetic complexity of the Discrete Fourier Transform from $O(n^2)$ to $O(n \log n)$. It is frequently described as “elegant” or “discovered rather than invented.” Within our framework, this phenomenology has a precise explanation.

The FFT’s recursive butterfly structure aligns simultaneously with: (a) the algebraic structure of the cyclic group $\mathbb{Z}/n\mathbb{Z}$ (algorithmic category structure); (b) cache-line-sized data access patterns in commodity DRAM (physical category structure); (c) data reuse patterns that minimize the interaction term Γ (energy profile structure).

The FFT is energetically coherent in the sense of Definition 4.2 because its algorithmic structure and its physical energy topology are nearly aligned: the morphism from $\mathcal{A}$ to $\mathcal{P}$ is nearly a category isomorphism, not merely a functor. The “discovered” phenomenology reflects the rarity of such alignment.

11.2 CPU Pipeline Architectures as High-Curvature Physical Categories

Modern CPU microarchitectures—deep pipelines, out-of-order execution, NUMA hierarchies—create a physical category $\mathcal{P}$ with what we call *high curvature*: the energy cost surface is highly non-local, with complex dependencies between physically distant morphisms.

In categorical terms: the physical category $\mathcal{P}_{\text{CPU}}$ is far from the “flat” category that the algorithmic category $\mathcal{A}$ assumes. The interaction term Γ is correspondingly large and highly context-dependent. Branch prediction corresponds to *speculative morphisms with rollback cost*: morphisms in $\mathcal{P}$ that have no counterpart in $\mathcal{A}$ and that contribute to Γ without contributing to any algorithmic structure.

The realization functor $F_{\text{CPU}} : \mathcal{A} \rightarrow \mathcal{P}_{\text{CPU}}$ is therefore highly lossy—far from the energetically coherent ideal of Definition 4.2.

11.3 Neuromorphic Computing as Categorical Collapse

Neuromorphic architectures [Mahowald and Douglas, 1991, Davies et al., 2018] take a different approach: rather than realizing algorithmic morphisms on a general-purpose physical substrate, they design the substrate so that the physical morphisms *are* the algorithmic morphisms. Spike-based communication implements computation and communication simultaneously; synaptic weights implement memory and processing simultaneously.

In our framework, this corresponds to an attempt to *collapse* F to near-identity: to design $\mathcal{P}$ so that $F(\mathcal{A}) \cong \mathcal{P}$. The advantage is minimal interaction term Γ and minimal energetic overhead.

The disadvantage—well-known to practitioners—is that G is correspondingly difficult to define. When physical morphisms are the computation, there are few stable high-level abstractions available: the profunctor $\mathbb{G}$ has many possible preimages and no canonical choice. Programmability and energetic efficiency are in adjoint tension, and neuromorphic systems currently live at one extreme.

12 Related Work

Resource-sensitive type systems. Linear types [Wadler, 1990] and their descendants—session types [Honda, 1993], graded modal types [Orchard et al., 2019], and quantitative type theories [Atkey, 2018]—provide type-theoretic mechanisms for tracking resource consumption at the language level. Our framework provides a categorical semantics for what such type systems are approximating at the physical layer.

Categorical semantics of computation. The interpretation of computational phenomena in terms of adjunctions has a long history: Lawvere’s work on comprehension [Lawvere, 1969], Moggi’s monadic semantics [Moggi, 1991], and the categorical semantics of linear logic [Seely, 1989] are foundational. Our contribution is to extend this tradition to the *physical* layer, connecting categorical structure to measured thermodynamic quantities.

Thermodynamics of computation. Landauer’s original paper [Landauer, 1961] and Bennett’s reversible computing programme [Bennett, 1973] establish the physical limits. Recent work by Wolpert and colleagues [Wolpert, 2019] on the stochastic thermodynamics of computation provides a modern framework that our categorical approach can be seen as abstracting.

Energy-aware computing. The systems literature on energy-aware computing [Barroso and Hölzle, 2007] typically treats energy as a scalar annotation on existing program representations. Our critique—that this is a forgetful functor without a right adjoint—is the formal counterpart to the practical observation that such annotations consistently fail to compose correctly across module boundaries.

Hardware-software co-design. The co-design literature [Patterson and Hennessy, 2018] advocates for joint optimization of hardware and software, which is adjacent to our proposal for a shared categorical formalism spanning both layers. Our contribution is the observation that such co-design requires not just joint optimization but a *shared adjunction*: the same categorical relationship must be legible at both the ISA level and the algorithmic level.

13 Toward an Experimental Programme

The framework is falsifiable at multiple levels.

Minimal compositional test. Select two morphisms f, g at the ISA level with independently measurable energy profiles. Measure $E(F(f))$, $E(F(g))$, and $E(F(f \circ g))$ over a distribution of execution contexts (as required by Section 9). Test whether

$$E(F(f \circ g)) = E(F(f)) \otimes E(F(g)) \otimes \Gamma$$

holds in a statistical sense, and estimate Γ as the residual. Vary compositional depth to identify the phase transition described in Section 7.

Adjunction unit measurement. For a given algorithm a , measure its physical realization $F(a)$, lift to the best available abstraction $G(F(a))$, and compare with a . The discrepancy is the unit η_a . Algorithms for which η is small are energetically coherent; those for which it is large are energetically incoherent. This gives a measurable, architecture-specific coherence score.

Landauer term isolation. Identify algorithmic morphisms of known injectivity class (injective, surjective, neither) and measure whether energy dissipation correlates with theoretical Landauer cost at the granularity of individual operations. This tests Proposition 8.2 directly.

14 Conclusion

We have developed a categorical framework in which energy is a first-class structural constituent of computation. The principal contributions are:

- (i) The identification of the realization functor $F : \mathcal{A} \rightarrow \mathcal{P}$ as the appropriate locus of the energy-algorithm relationship, and the demonstration that current realizations are energetically forgetful.
- (ii) The energetically coherent adjunction $F \dashv G$ as the correct structural ideal, with the unit and counit providing formal content to the notion of “optimal tension” between algorithmic and physical structure.
- (iii) The profunctor generalization of the abstraction functor G , motivated by the generic non-uniqueness of the lifting from physical measurement to algorithmic structure.
- (iv) The analysis of the interaction term Γ and its phase transition from perturbative to dominant at large compositional depth, explaining the failure of additive energy models at scale.
- (v) The morphism-level entanglement of energetic and information-theoretic structure via Landauer’s principle, and its implication that a cost-preserving functor must track entropy production per morphism.
- (vi) The bootstrapping problem in fine-grained energy measurement, and its resolution via contextual energy distributions.
- (vii) The thermodynamic redefinition of correctness as joint satisfaction of behavioral specification and energetic coherence within adjunction-determined tolerance.

The framework unifies several currently disparate research fronts—linear dependent type theory, certified compilation, hardware-software co-design, and the thermodynamics of computation—by revealing them as different projections of the same underlying adjunction. This convergence suggests that progress on any one front is likely to yield corresponding progress on the others, and that the most leveraged research direction is the one that addresses their shared root: the formal, measurable, composition-preserving relationship between algorithmic structure and physical energy.

References

Atkey, R. (2018). Syntax and semantics of quantitative type theory. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 56–65.

- Barroso, L. A., and Hölzle, U. (2007). The case for energy-proportional computing. *Computer*, 40(12):33–37.
- Bennett, C. H. (1973). Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532.
- Cooley, J. W., and Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.
- Davies, M., et al. (2018). Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99.
- Girard, J.-Y. (1987). Linear logic. *Theoretical Computer Science*, 50(1):1–101.
- Honda, K. (1993). Types for dyadic interaction. In *Proceedings of CONCUR '93*, Lecture Notes in Computer Science, vol. 715, pages 509–523. Springer.
- Kocher, P., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *Advances in Cryptology—CRYPTO '99*, Lecture Notes in Computer Science, vol. 1666, pages 388–397. Springer.
- Landauer, R. (1961). Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5(3):183–191.
- Lawvere, F. W. (1969). Adjointness in foundations. *Dialectica*, 23(3–4):281–296.
- Lorègian, F. (2021). *(Co)end Calculus*. London Mathematical Society Lecture Note Series, vol. 468. Cambridge University Press.
- Mac Lane, S. (1998). *Categories for the Working Mathematician*, 2nd ed. Graduate Texts in Mathematics, vol. 5. Springer.
- Mahowald, M., and Douglas, R. (1991). A silicon neuron. *Nature*, 354:515–518.
- Moggi, E. (1991). Notions of computation and monads. *Information and Computation*, 93(1):55–92.
- Orchard, D., Liepelt, V., and Eades III, H. (2019). Quantitative program reasoning with graded modal types. *Proceedings of the ACM on Programming Languages*, 3(ICFP):110.
- Patterson, D., and Hennessy, J. (2018). *Computer Organization and Design RISC-V Edition*, 2nd ed. Morgan Kaufmann.
- Riehl, E. (2017). *Category Theory in Context*. Dover Publications.

- Seely, R. A. G. (1989). Linear logic, *-autonomous categories and cofree coalgebras. In *Categories in Computer Science and Logic*, Contemporary Mathematics, vol. 92, pages 371–382. American Mathematical Society.
- Wadler, P. (1990). Linear types can change the world! In *Programming Concepts and Methods*, pages 546–566. North-Holland.
- Wolpert, D. H. (2019). The stochastic thermodynamics of computation. *Journal of Physics A: Mathematical and Theoretical*, 52(19):193001.